Typed CPUおよびTyped OSのアイディア

住井 英二郎
東京大学大学院 情報学環
米澤研究室

背景: 型システムの役割の拡大

型付きプログラミング言語 (ML等)

- プログラマの誤謬を防止

→ 型付き中間コード (TIL, ML Kit 2等)

- 最適化やメモリ管理に利用

→ 型付きアセンブリ/バイトコード (TAL, JVML等)

- 安全性を保障

背景: 型システムの役割の拡大

型付きプログラミング言語 (ML等)

- プログラマの誤謬を防止

→ 型付き中間コード (TIL, ML Kit 2等)

- 最適化やメモリ管理に利用

→ 型付きアセンブリ/バイトコード (TAL, JVML等)

- 安全性を保障

→ 型付きCPU? 型付きOS?

アイディアその1: "Typed CPU" CPUの内部で型情報を利用

- レジスタの値がアドレスか整数か
- メモリの値が「定数」か否か
- Region情報によるalias解析 (cf. TAL with regions)
  → Aggressiveなout-of-order実行
- Loadする値の（擬似）線形性
  → Cachingの省略

etc.

アイディアその2: "Typed OS" ソフトウェアによる保護

「OS(というかMMU)による保護はオーバーヘッドが小さい」の嘘

- CPUの中で大きなオーバーヘッドを払っている！(はず)
  - クリティカルパス長, 消費電力等

アイディアその2: "Typed OS" ソフトウェアによる保護

「OS(というかMMU)による保護はオーバーヘッドが小さい」の嘘

- CPUの中で大きなオーバーヘッドを払っている！(はず)
  - クリティカルパス長, 消費電力等

⇒ 型安全なバイナリしか実行させないシステムなら不要に？
  - 静的検査 + SFI

課題: どうやって実証するか？

- 本当にCPUやOSを0から開発する？
  - 既存のシステムとどのように比較するか？
- とりあえずLinuxカーネルに（制限した）OCamlランタイムを組み込む等する？

「ネタ」リスト

- VHDL + FPGAでシンプルなCPUを作る（MIPS等のユーザレベル）
  - one-chip multi-processorなどへ
- Linuxカーネル等に安全な言語の実行系を組み込む
  - TAL, Java, OCaml, etc.
- OCaml等のIA-64用のコンパイラを作る

などなど